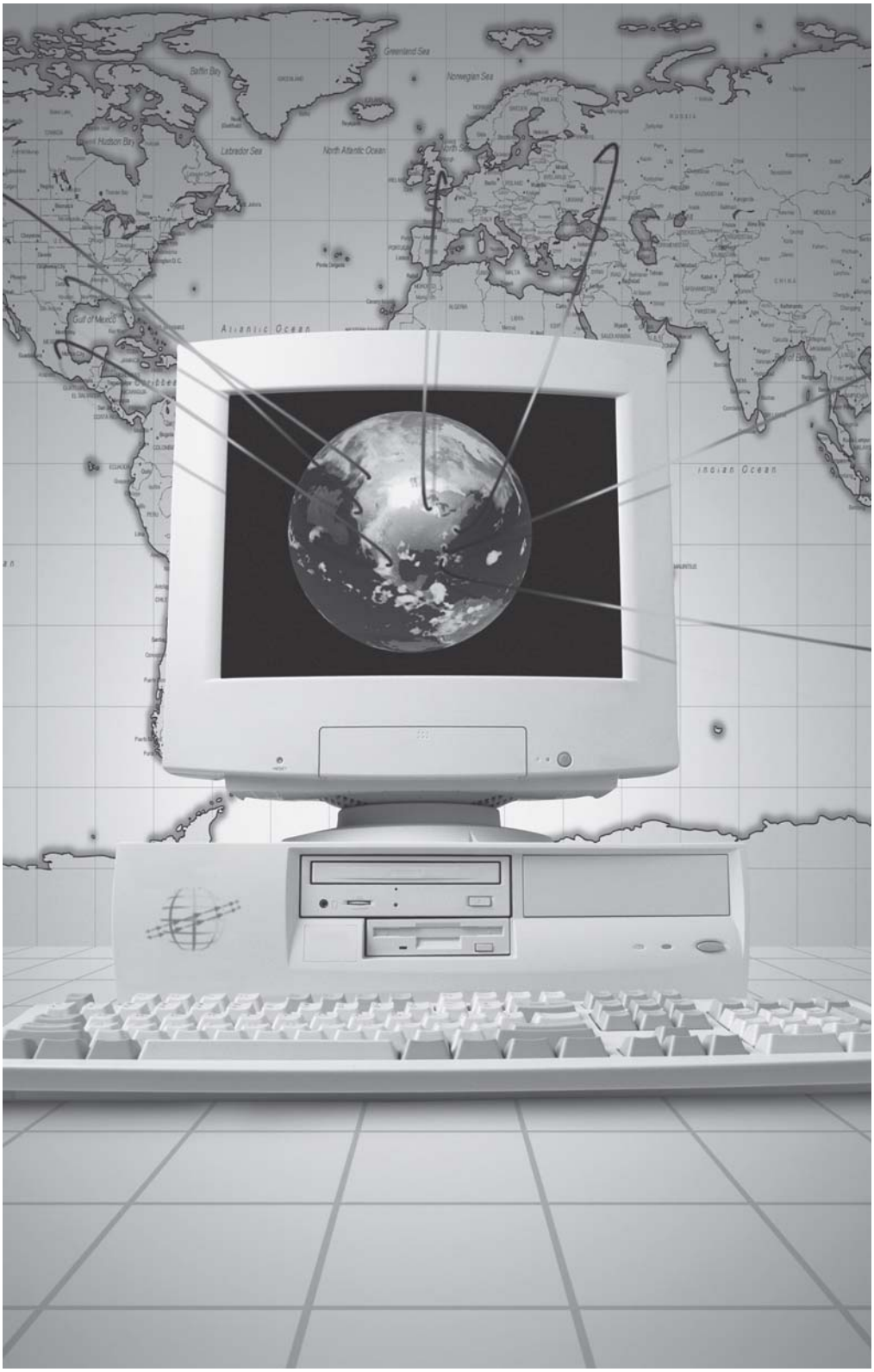
Greenland Sea
Baffin Bay
Norwegian Sea
Hudson Bay
Labrador Sea
North Atlantic Ocean
Atlantic Ocean
Indian Ocean

NET-CENTRIC WARFARE AND ITS IMPACT ON SYSTEM-OF-SYSTEMS

LT COL STEVEN G. ZENISHEK, USAF
AND DR. DAVID USECHAK

The effects of Net-Centric Warfare (NCW) and its impact on the acquisition of System-of-Systems constructs as experienced by the acquisition of the Air Force Distributed Common Ground System Block 10.2 are examined. Block 10.2 is an Acquisition Category III program that is fielding a net-centric, service-oriented architecture for intelligence, surveillance, and reconnaissance. The NCW links sensors, communications systems, and weapons systems in an interconnected grid creating seamless data and information flows to warfighters, policy makers, and support personnel. The Office of the Assistant Secretary of Defense Networks and Information Integration has identified net-centric best practices as using a Service-oriented Architecture, implementing a data-centric strategy, information assurance strategy and use of Net-Centric Operations Warfare Reference Model. Block 10.2 has found these best practices shift the acquisition strategy from acquiring a System-of-Systems to acquiring an enterprise of services.

This is an examination of the effects of Net-Centric Warfare (NCW) and its impact on the acquisition of System-of-Systems (SOS), as experienced by the Air Force Distributed Common Ground System (AF-DCGS) Block 10.2 program. The AF-DCGS is a globally dispersed, wide area network composed of fixed and mobile ground processing systems for Intelligence, Surveillance, and Reconnaissance (ISR) data from manned and unmanned aerial vehicles, satellites, and ground sensors. The AF-DCGS receives intelligence feeds that are processed, stored, correlated and fused, exploited, and disseminated to Air Operations Centers (AOCs) for strike planning and execution, and to support Joint Task Force Commanders. Block

10.2 is the Air Force's first major ISR upgrade. Its goal is to replace the current ISR systems with a net-centric ISR enterprise that provides services to any warfighter requiring ISR information.

The NCW theorizes that an enterprise composed of interconnected nodes increases mission effectiveness exponentially as the number of nodes increases linearly. The NCW links sensors, communications systems, and weapons systems in an interconnected grid creating seamless data and information flows to warfighters, policy makers, and support personnel. The NCW is based on the tenets that networked forces:

- Effectively improve information sharing.
- Greatly enhance the quality of information and shared situational awareness (Shared situational awareness enables collaboration, self-synchronization, and enhances sustainability and speed of command).
- Dramatically increase mission effectiveness.

Current SOS integration methodologies are inadequate for implementing a net-centric enterprise. The focus of SOS integration is on integrating complete systems using point-to-point integration, Enterprise Application Integration (EAI), or Business Process Management (BPM). These methodologies have significant limitations. Point-to-point integration is extremely sensitive to minor changes to the interface. Typically, each interface is a custom integration using proprietary messages or application program interfaces (APIs). A point-to-point interface requires *tight coupling*, resulting in a lack of *agility*, slow acquisition times, and high cost to change. In other words, each system *talks* to other systems on a one-to-one basis using messages in exact formats.

These interface types are fragile because they will fail anytime a change is made to either system. The difficulties encountered with point-to-point interfaces resulted in the use of EAI middleware. However, interfaces developed using EAI middleware also resulted in fragile proprietary implementations, which are costly and time intensive to change. The BPM was introduced to increase the performance (return on investment, profits, and so forth) of business units; however, BPM proprietary software implementations also resulted in fragile implementations. Therefore, a net-centric enterprise must be implemented using open standards, non-proprietary APIs, *loose coupling* between data and applications, and agile (i.e., not fragile) interfaces.

NEW CONSTRUCT: AN ENTERPRISE OF SERVICES

The challenge to the acquisition community is how to implement a net-centric enterprise. A common misconception is that a Web-based application is automatically a net-centric application, but this misconception results in a myopic focus on technology. A net-centric enterprise approach requires the stakeholders to focus on the whole enterprise as an integrated architecture to include work processes, data flows,

network communications, Web services, and so forth. A net-centric enterprise requires a software integration framework to separate data from applications by managing loosely coupled software interactions called services.

Modeling the business processes (i.e., Concept of Operations [CONOPS]) is critical to ensuring the services *know* with whom and when data are shared on a many-to-many basis. A robust information assurance strategy and processes are required to ensure only authorized persons are provided information services—and not the "bad guys." The result is the enterprise is no longer system centric, but it is transformed to providing information services to stakeholder subscribers in the enterprise. These information services include, but are not limited to, messaging, information discovery, mediation, collaboration, data storage, applications (software that manipulates the data), and information assurance. The result is that you quickly discover you are building an enterprise of services, which is developed and managed radically differently from a system-of-systems.

NET-CENTRIC ATTRIBUTES

The Office of the Assistant Secretary of Defense Networks and Information Integration (OASD NII) has developed a net-centric checklist to assist program managers in understanding net-centric attributes (The Office of the Assistant Secretary of Defense Networks and Information Integration [OASD NII], 2004). The attributes include a System-Oriented Architecture (SOA), net-centric data strategy, information assurance strategy, and the Net-Centric Operations Warfare (NCOW) reference model.

The first attribute, an SOA, is an architecture made up of components (software) and connections in which interoperability and location transparency are key attributes. The SOA is really about development, design, and integration (i.e., the building of the system) of software components that are addressable by a heterogeneous network. Stated another way, an SOA is a framework of software technologies designed to support interoperable component-to-component interactions over a network.

One of the major issues facing legacy systems is the difficulty in making changes to the system because of the tightly coupled design. The SOA solves the tight coupling design problem by providing a loosely coupled design through the use of open standards that masks the underlying technical details. Services are allowed to asynchronously (i.e., independently) access data, business processes, and infrastructure. The agility in which the interface tolerates changes and allows many-to-many exchanges is the result. The SOA also solves the unique, and in some cases proprietary, interfaces by using currently accepted open standards to define interfaces. The SOA implements a layered architecture for reuse of existing systems and applications, and transforms them into agile information services. The Net-Centric Checklist lists SOA best practices as:

- Design application and system functionality as accessible and reusable services.
- Expose service functionality through open standard interfaces.

- Maintain an abstraction layer between service interfaces and service implementations.
- Describe service interfaces using standard metadata.
- Advertise and discover services using standard service registries.
- Communicate with information services using standard protocols (OASD NII, 2004).

The second attribute of Net-Centric Data Strategy is aimed at making data available when and where needed (see Figure 1). The elements of this strategy include:

- Make data visible, available, and usable when needed and where needed for accelerated decision making.
- *Tag* all data to enable discovery of data by users.
- Post all data for all users to access except when limited by security, policy, or regulations.
- Enable many-to-many exchanges in a network environment (OASD NII, 2004).

The need to post all data and provide access for all users requires a robust information assurance strategy. The Net-Centric Checklist describes an information assurance strategy as: an integrated Identity Management, Permissions Management, and Digital Rights Management that ensures adequate confidentiality, availability, and integrity.

The Net-Centric Checklist refers to the Net-Centric Operations Warfare Reference Model as "the target viewpoint of the Department's Global Information Grid. This viewpoint is a service-oriented, inter-networked, information infrastructure in which users request and receive services that enable operational capabilities across the range of (1) military operations, (2) DoD [Department of Defense] business operations, and (3) Department-wide enterprise management operations. As programs plan, the Reference Model must be included in the program planning" (OASD NII, 2004, p. i).

AIR FORCE DISTRIBUTED COMMON GROUND SYSTEM (AFDCGS) BLOCK 10.2

The Air Force Distributed Common Ground System (AFDCGS) Block 10.2 is an Acquisition Category (ACAT) III acquisition program that is fielding a net-centric, service-oriented architecture for ISR with Combat Operations Command and Control. Block 10.2 was initially planned as part of an evolutionary acquisition strategy in which a block is defined as "a militarily useful and supportable operational capability

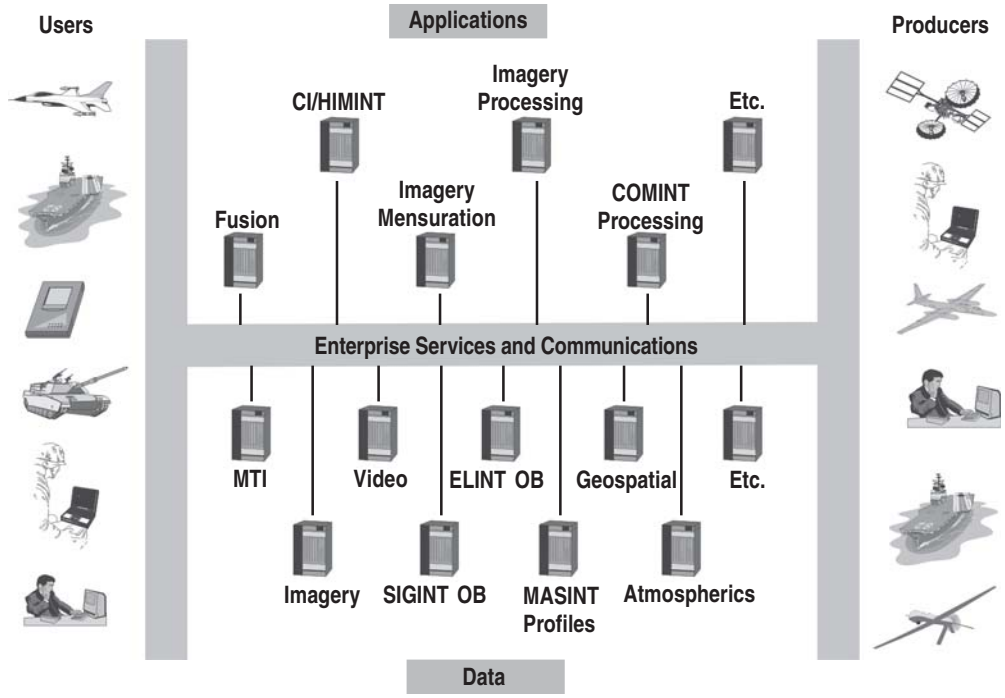

FIGURE 1. DOD VISION OF NET-CENTRIC SERVICES

that can be effectively developed, produced or acquired, deployed, and sustained" (Hawthorne & Lush, 2002, p. 14).

Block 10.2 acquisition was initiated in August 2002 and has gone through requirements definition, market research, competitive source selection, Defense Acquisition Board (DAB), contract award, architecture design, associated software builds, and will be installed at the Transformation Center, Langley AFB, Virginia in June 2005 to begin formal development and operational testing—all within 34 months. The OASD Acquisition, Technology, and Logistics (AT&L) directed a DAB be convened to settle the issue on whether to implement common ISR systems for all Services (i.e., military departments), or the implementation of a Service-oriented architecture for ISR. The DAB chair authorized Block 10.2 to implement a service-oriented architecture approach. All the military services agreed to collaborate with the Air Force in defining a common backbone infrastructure to ensure interoperability.

While Block 10.2 was initially conceived as an evolutionary acquisition, it is really a revolutionary or disruptive change acquisition. The revolutionary acquisition approach implemented many acquisition methodologies that have proven critical to the success of a net-centric enterprise: for example, architecture design, integration framework, universal modeling language (UML), standards-based acquisition, lean development, agile acquisition, and metadata management.

The AFDCGS Block 10.2 Enterprise was designed using the Department of Defense (DoD) Architecture Framework (DoDAF). The contractor was required to develop

architectural views as defined by the statement of work. A typical architecture development can take years, but the majority of Block 10.2 architecture was completed within 5 months after contract award! This can be explained by the synergy of a number of factors. First, the government identified required net-centric standards within the Block 10.2 technical requirements document (TRD). Second, the government limited the architecture to implementing those capabilities required within a 3-year period. Third, the government provided the existing operational architecture views to the contractor. Fourth, the government specified that the contractor present DoDAF architecture views at the initial and final design reviews. Finally, the contractor team was highly experienced with Air Force ISR environments. The result was the contractor architects focused on solving a specified problem set versus trying to define a perfect implementation for the next 20 years.

The architecture lesson learned was that very few government personnel, during the design reviews, understood what they were reviewing. Almost all government participants expected to see a traditional design specification to include data *threads*. Instead, the architecture design defined where and how data flowed, but did not show a complete design because commercial off-the-shelf (COTS) products within the integration framework already provided 80 percent of the services. Therefore, the Block 10.2 architecture design was actually the process of defining integration activities and the configuration of the COTS products to provide services.

One of the current challenges is how to implement SOA while at the same time capturing capabilities (services) from legacy systems.

Block 10.2 is built upon an integration framework, which provides loosely coupled services. The Block 10.2 framework is an SOA based on Java 2 Enterprise Edition (J2EE) specification to provide loosely coupled Web services. This framework is called the DCGS Integration Backbone (DIB). Much confusion exists about what the DIB actually is. The DIB is composed of tools, standards, architecture, documentation, and software. It provides established patterns to help client applications connect to a service. As a result, the DIB is scalable and extensible ranging from a workstation configuration to a high-end server farm. It is important to note that the DIB implements identified standards—it does not create new standards. A lesson learned is that architecture design is required to define how to configure the DIB to meet operational requirements. Therefore, sound systems engineering is required—not to define a system, but to design services required to meet warfighting requirements.

One of the current challenges is how to implement SOA while at the same time capturing capabilities (services) from legacy systems. The AFDCGS contractor team came up with a construct to explain how a system could be integrated with the DIB

to provide or become an information service. It is important to note these types of integration do not correspond to levels of compliance as used by Defense Information Infrastructure Common Operating Environment (DII COE), Level of Information system Interoperability (LISI), or any other definition or compliance levels. Instead, these are types of systems integrations that are done based on business process engineering, cost, schedule, and performance needs. One type of integration is no better than another.

- Type 0 integration is basically point-to-point messaging.
- Type 1 integration *wrappers* the application's database to expose data to the enterprise (see Figure 2).
- Type 2 integration *wrappers* the application to expose it to the enterprise. It enables workflow management for the application within a site (see Figure 3).
- Type 3 integration refactors the application into the DIB while wrappering the applications database. This allows the application to be distributed among multiple sites for distributed operations (see Figure 4).
- Type 4 integration fully refactors the application into the DIB or a new application is built into the DIB. This is an ubiquitous application that exists within the enterprise and can be used across multiple sites or enterprises (see Figure 5).

Net-centricity begins with Type 2 integration because it allows the enablement of many-to-many interactions.

The UML was critical to the development of Block 10.2 applications also known as the Multi-intelligence core, which processes the data resident in the DIB. The UML models were developed collaboratively between the contractor, program office, and warfighter subject matter experts (SMEs) to define interactions between applications and the DIB. The UML complimented the architecture design by defining the workflow patterns necessary to implement distributed operations between multiple sites. Our lesson learned was that a collaborative UML process with the user greatly enhanced the contractor's understanding and ability to meet the aggressive design schedule.

Block 10.2 net-centric paradigm is a standards-based acquisition. As such, a DCGS Acquisition Standards Handbook—Imagery (DASH-I) has been developed to identify the standards required to implement the DIB and the Multi-Intelligence Core applications. In addition, a set of standards has been identified for imagery, and these can be found in the DASH-I. Our lesson learned is that open standards are critical to ensuring integration.

Lean programming was used to great effect to achieve agility (Poppendieck, 2001). It has 10 tenets:

1. Eliminate waste.

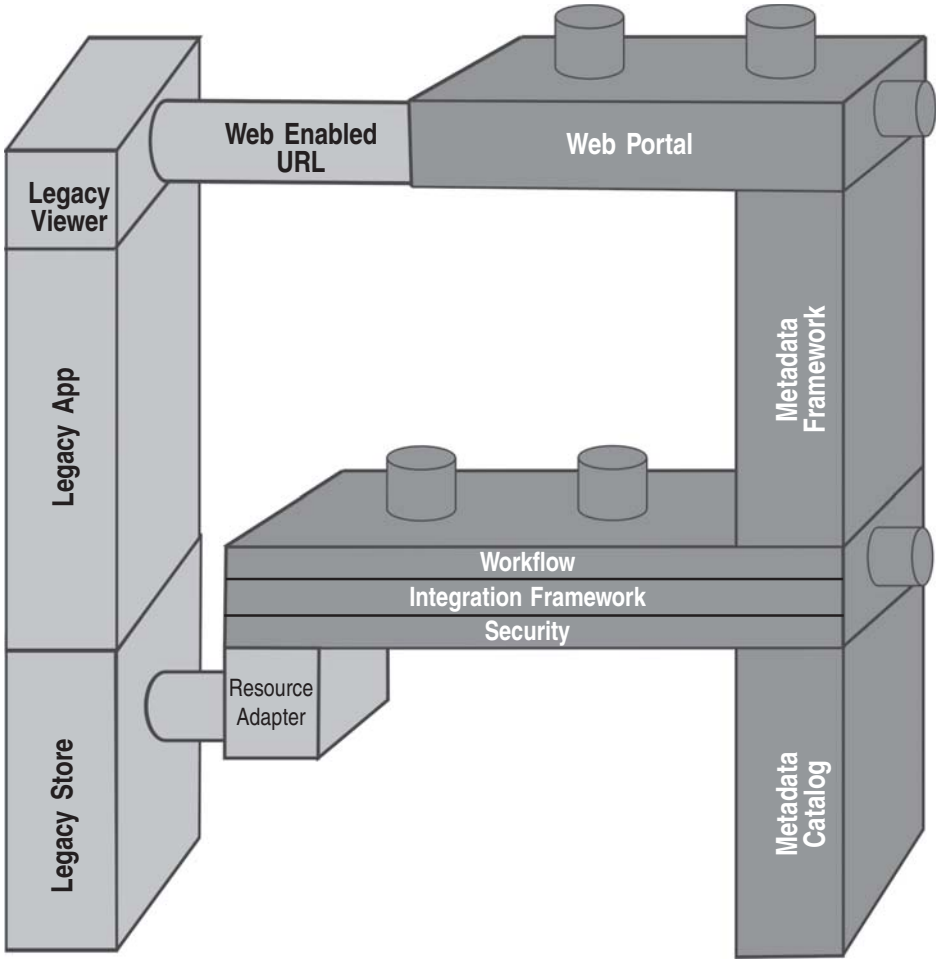

FIGURE 2. TYPE 1 INTEGRATION: WRAPPER LEGACY DATA STORE

2. Minimize paperwork.
3. Implement in small increments.
4. Decide as late as possible.
5. Decide as low as possible.
6. Satisfy all stakeholders.
7. Focus on testing.
8. Measure business results.

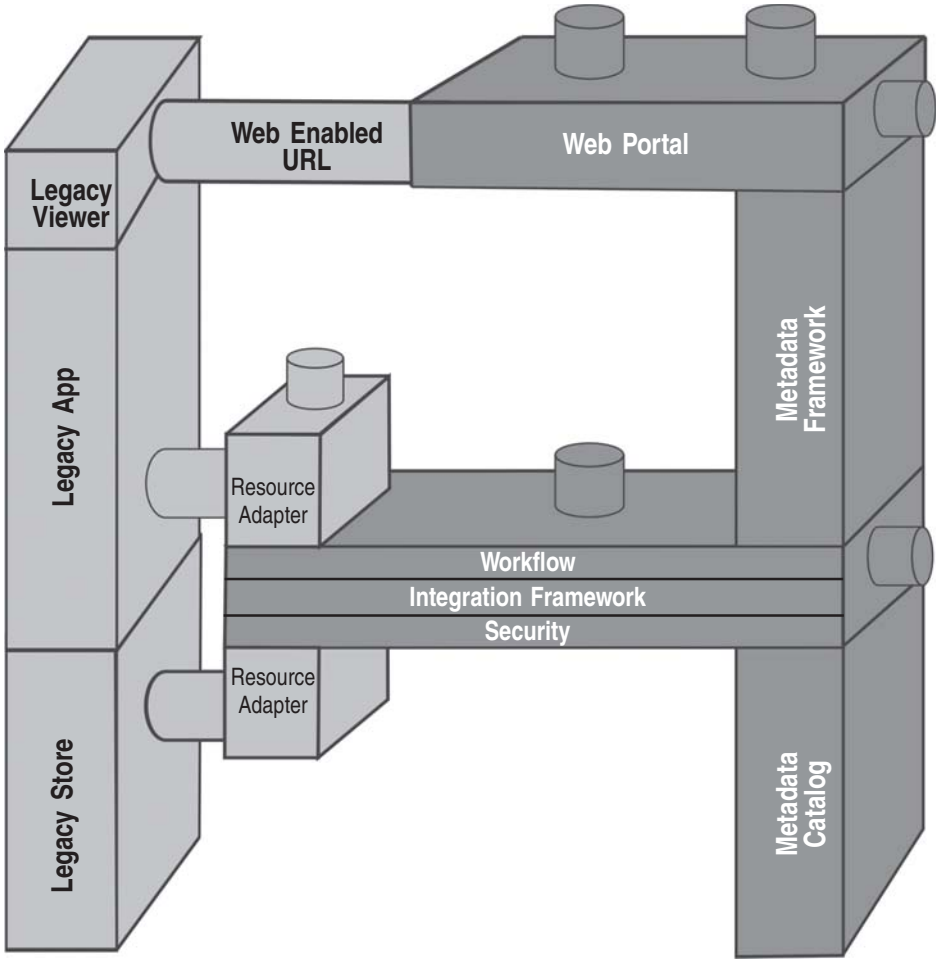

FIGURE 3. TYPE 2 INTEGRATION: WRAPPER LEGACY APPLICATION AND DATA STORE

9. Optimize across organizations.

10. Never stop improving.

Block 10.2 was lean by focusing on delivering well-defined capabilities within three years, and as a result, the program was tailored to achieve this objective. In addition, the architecture design documents the enterprise of services and replaces the function of a specification. Acquisition agility—the need for change to define, not disrupt, the program—was required because this technology experiences changes every six months. As a result, spirals were structured to provide new capabilities every six months and allow change as late as possible.

Decision-making authority was forced as low as possible. For example, the source selection authority was delegated to a colonel level. Weekly *telecoms* were established

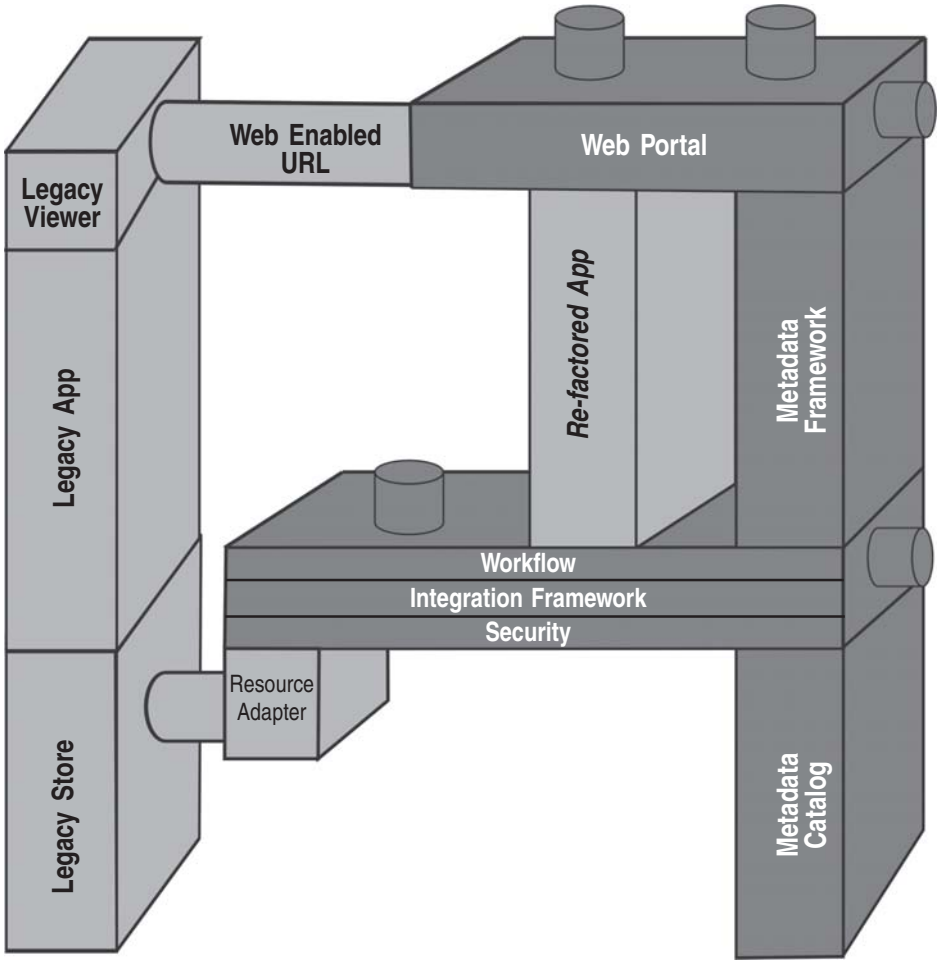

FIGURE 4. TYPE 3 INTEGRATION: REFACTORED APPLICATION

with stakeholders to provide information and to ensure satisfaction. The UML process allowed the developers to develop code based on passing pre-defined test cases. Schedule progress, software metrics, and earned-value metrics were tracked to measure contractor performance. Monte Carlo Modeling was used to assess confidence in the contractor's schedules. As a result, the program office never had any surprises in contractor performance. Our lesson learned is that lean programming enhanced the program office's ability to focus on product and deliver results.

Acquisition agility can be described as using change to define the program versus letting change disrupt the program. Agility was achieved by the synergy of architecture design, integration framework, UML, standards-based acquisition, and lean development. Each of these practices brings with it an element of agility. The lesson learned is that acquisition agility allows the program office to shorten its decision-making loop and rapidly respond to changes in the environment. Block 10.2 has proven

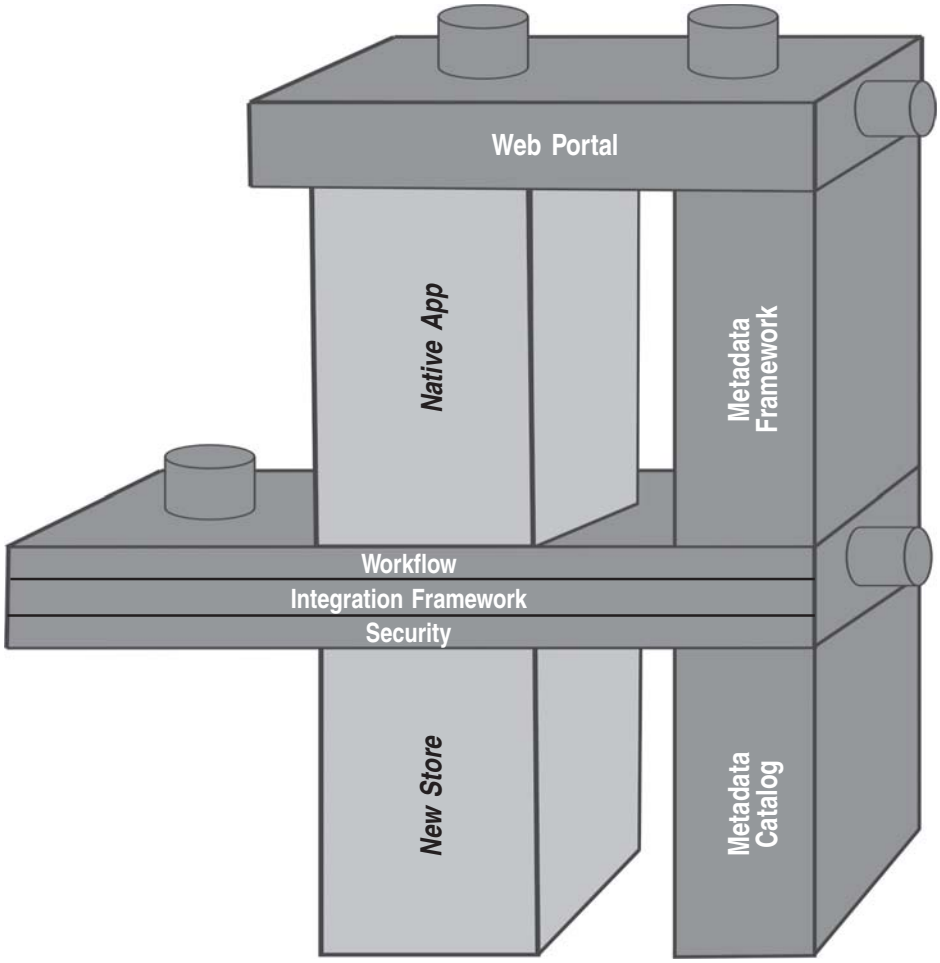

FIGURE 5.
TYPE 4 INTEGRATION: UBIQUITOUS APPLICATION AND DATA STORE

itself agile by its ability to run multiple events in parallel while responding to major unanticipated events. For example, the program office found itself simultaneously preparing for a DAB while conducting a major source selection, and successfully completed both within five months. Then the program office found itself rapidly executing a major program while simultaneously responding to a General Accounting Office, now Government Accountability Office (GAO), protest—both were also successfully completed.

The program office faced yet another challenge, metadata management. In an SOA, data are identified and catalogued for discovery within the metadata catalog. Metadata has worked for over a quarter of a century, but Block 10.2 is implementing one of the first operational metadata catalogs. The program office discovered that metadata registries exist to register data for discovery, but no one is managing the metadata definitions. For example, there are over 54 definitions of latitude and longitude in

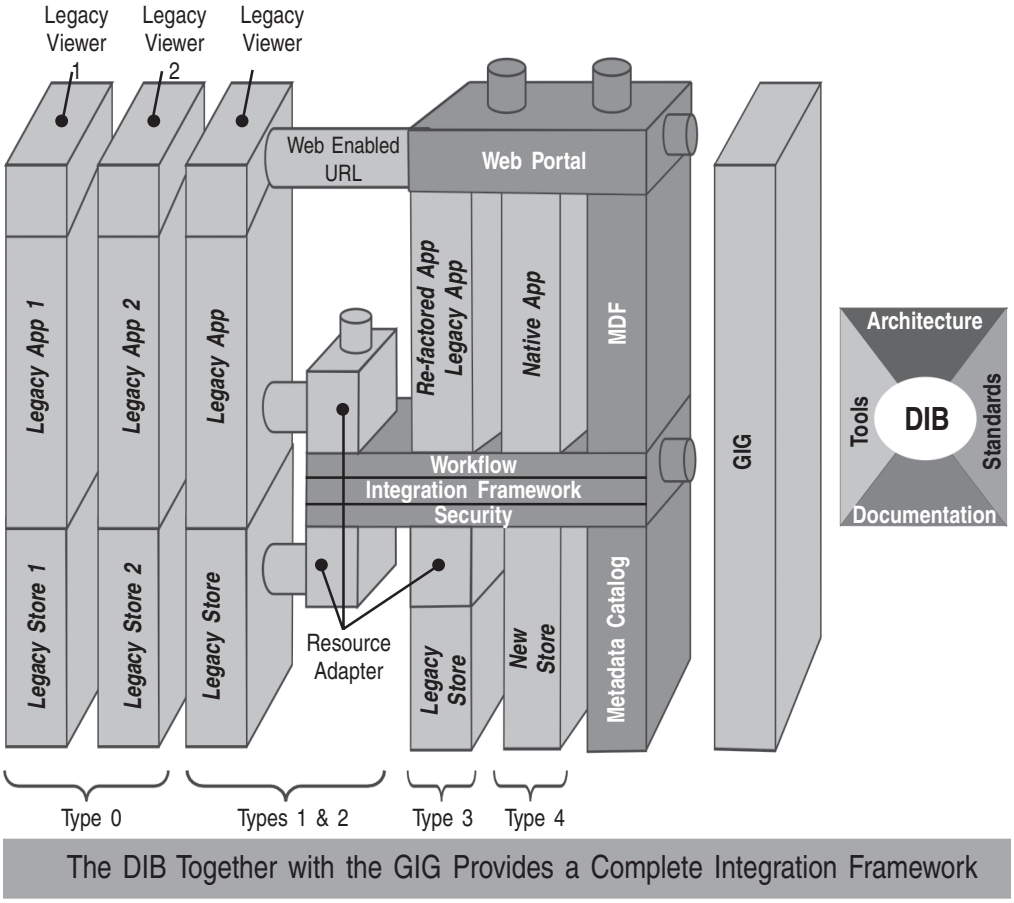

FIGURE 6. PUTTING IT ALL TOGETHER

the DoD registries, but AF DCGS only needs a subset of these. In addition, the multiple registries are not harmonized. Therefore, the contractor is implementing only those required as defined by the architecture and UML modeling. The lesson learned is that program offices will have to perform extensive metadata management and harmonization within its enterprise until DoD implements a metadata management process.

AREAS REQUIRING RESEARCH

Net-centric programs require new ways of management which the current program management practices have not anticipated. Net-centricity not only affects the acquisition process, but it also affects requirements, planning, programming, budgeting, CONOPS, doctrine, training, systems engineering, etc. It requires new acquisition concepts, such as how to perform cost-estimating and how to award contracts. As a result, program management must also transform to become net-centric. Program

offices will be required to restructure to operate in an enterprise in real time. In order to achieve this—new management, acquisition processes, and procedures need to be established.

Universal Description, Discovery, and Integration (UDDI) was originally developed to allow commercial industry to discover each other's services. In the new world of net-centric service-oriented architectures, the focus of program offices shifts from developing systems to developing, integrating, and subscribing to services. To avoid duplication of effort and cut costs, there exists a need for program offices to post what services they are developing so others can subscribe. It will also require the modification of services to meet other program office's needs. The result will be a shift in the requirements and system engineering processes. The shift will be from systems development to identifying requirements, determining if services already exist that meet the requirement, and if not, modifying an existing service or developing a new service and registering it for others to use. This will enable self-synchronization between programs. However, unlike combat operations, acquisition lacks a published *commander's intent* requiring program offices to apportion, collaborate, and synchronize. Therefore, the acquisition community must transform to enable discovery, self-synchronization, and apportionment of services. Research is needed to identify options for making this a reality.

Actual experience to date is that while software code has grown 60 percent, cost and schedule has increased only 18 percent (i.e., about 10 weeks).

Cost estimating will become more complicated. Return on investment will focus on whether there are existing services, services that can be modified, or if a new service is required to be developed. In many cases, several alternative services may co-exist until user feedback can determine if a "best of breed" or best value exists. The challenge will be how to cost a service, since it is different from the current system cost models. Another complicating factor is that the framework used for Block 10.2 includes automated software coding tools. For example, coders now can specify what task the Web service needs to provide: they click a command, and the framework tool automatically generates the code and even notifies them if they violated any rules in specifying the task in seconds. As a result, code is generated much faster with far less errors. Block 10.2 initially used the Constructive Cost Model (COCOMO) to evaluate cost and schedule risk. The COCOMOs predicted Block 10.2 would take 6 to 18 months longer than the planned 15 months to factory acceptance test. Actual experience to date is that while software code has grown 60 percent, cost and schedule has increased only 18 percent (i.e., about 10 weeks). Therefore, COCOMO is a far too conservative model for cost estimating when using a J2EE framework for several

reasons. First, J2EE modularity allows modules to be completed in parallel while the COCOMO assumes serial development. Second, some J2EE frameworks now incorporate automatic code generation tools while the COCOMO assumes manual generation. Finally, J2EE framework allows extensive code reuse, which COCOMO does not fully recognize. Other costing models might have similar problems and therefore they all will require updating to handle this new acquisition paradigm.

A lesson learned using J2EE framework is that the real schedule and cost driver is integration, not software. We have learned that it takes about 30 percent longer to resolve an integration discrepancy versus a software discrepancy. The reason is an error in software can be traced to a particular line of code. An integration "error" is the result of a mismatch between different COTS/GOTS vendor products and is much harder to isolate. We had a low number of discrepancies for an effort of this size and complexity, but isolating the cause of those discrepancies took longer. We also suspect that we uncovered a number of latent problems that currently exist in the field, but were unreported until we integrated separate components into a seamless Enterprise Architecture.

An Enterprise of Services is a new acquisition paradigm for operating within a net-centric environment.

Another area requiring research is how to fund the development, implementation, and sustainment of services that includes the unanticipated user. For example, in the commercial industry, Web services are paid through advertisements, fee for service, and *pop-ups*. In the DoD, the user funds the program office to meet its requirements. In an Enterprise of Services, anyone with the need to know can call up the service. This will drive new requirements for computing power to support more users than the original customer has budgeted. The question becomes "Who pays?" The experience of Block 10.2 has been that everyone is trying to leverage the Air Force program, but no one other than the Air Force has funded any capability. Therefore, a fair and just process must be implemented that permits the continued development, implementation, and support of fielded information services across DoD.

CONCLUSION

An Enterprise of Services is a new acquisition paradigm for operating within a net-centric environment. It requires the use of an SOA that separates the applications from the data and allows discovery of data and information services. A robust information assurance strategy is a must.

An Enterprise of Services approach is a revolutionary acquisition process. The focus of the acquisition becomes identifying services to meet user requirements with an emphasis on many-to-many interfaces. It requires a disciplined acquisition process that includes the use of architecture design, integration framework, UML, standards-based acquisition, lean programming, agile acquisition, and metadata management.

However, current acquisition management practices are not designed to operate in a net-centric environment. Discovery of other program offices' efforts needs to be implemented to allow subscription and self-synchronization. Cost models need to be updated to reflect the new software methodologies now being implemented that allow significant cost and schedule savings. Finally, new methods of funding information services need to be investigated, as the current practice of a sole user footing the bill is unfair in an era of supporting unanticipated users.

ACKNOWLEDGMENT

The authors would like to make the following acknowledgment: Figures 2 through 6 courtesy of Raytheon.

Lieutenant Colonel Steven G. Zenishek, USAF, is a graduate of DSMC (PMC 99-2). His program management career includes acquisition of airlift C2 (AMC C2 IPS), Theater Battle Management Core Systems (TBMCS), assignments in Air Force PEO office, and Assistant Secretary of the Air Force for Acquisition (SAF/AQI). He currently is the program manager for DCGS Block 10.2, has a bachelor's degree in engineering from Iowa State University, a master's degree from the University of North Dakota, and Project Management Professional (PMP) certification from the Program Management Institute (PMI). He is also a 1999 graduate of DAU (Advanced Program Management Course).

(E-mail address: steven.zenishek@hanscom.af.mil)

Dr. David Usechak has 35 years' experience in program management, engineering, software and system development, integration and testing of command, control, communication, computers, intelligence, surveillance, and reconnaissance (C4ISR) systems for the U.S. Army. Usechak received his D.Sc. in engineering from the New Jersey Institute of Technology, a master's degree in aerospace and mechanical science from Princeton University, a master's degree in electrical engineering from Fairleigh Dickinson University, and a bachelor's degree in electrical engineering from Widener University. He is a 1991 graduate of DAU.

(E-mail address: dusechak@osec.com)

AUTHOR **BIOGRAPHY**

REFERENCES

BEA Education Services. (2004). *BEA WebLogic platform 8.1: Workshop on service-oriented architectures.* San Jose, CA: BEA Systems, Inc.

Deputy Secretary of Defense for Networks and Information Integration, Department of Defense, Deputy Chief Information Officer. (2003, June 2). *DoD discovery metadata standard (DDMS) (Version 1.2).* Retrieved November 27, 2004, from http://www.afei.org/news/ddms.pdf

Electronics Systems Center/SRHG. (2003). *AF DCGS Block 10.2 DCGS integration backbone technical requirements document.* Hanscom AFB, MA: Department of the Air Force, ESC.

Hawthorne, S., & Lush, R. (2003). Evolutionary acquisition and spiral development. *Crosstalk*. The Journal of Defense Software Engineering. Hill AFB, UT: Software Technology Support Center, Department of the Air Force.

Office of the Assistant Secretary of Defense for Networks and Information Integration, Department of Defense, Chief Information Officer. (2004, February 9). *DoD architecture framework* (Version 1.0). Retrieved November 27, 2004, from http://www.defenselink.mil/nii/doc/DoDAF_v1_Volume_I.pdf

Office of the Assistant Secretary of Defense for Networks and Information Integration, Department of Defense, Chief Information Officer. (2004, May 12). Net-centric checklist (Version 2.1.3). Retrieved November 27, 2004, from http://www.defenselink.mil/nii/org/cio/doc/NetCentric_Checklist_v2-1-3_May12.doc

Poppendieck, M. (2001, May). Lean programming. *Software Magazine.* Retrieved November 27, 2004, from http://www.poppendieck.com

Universal Description, Discovery, and Integration (UDDI) 2.0. (June 18, 2001). Retrieved November 27, 2004, from http://www.uddi.org and http://www.oasis-open.org/committees/uddi-spec